applied sciences

MDPI

Article

Reversible Data Hiding in Encrypted Color Halftone Images with High Capacity

Yu-Xia Sun [1], Bin Yan [1,*], Jeng-Shyang Pan [2], Hong-Mei Yang [2] and Na Chen [1]

[1] College of Electronic and Information Engineering, Shandong University of Science and Technology, Qingdao 266590, China; sunyxsdust@outlook.com (Y.-X.S.); na_chen_xd@126.com (N.C.)
[2] College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China; jspan@kuas.edu.tw (J.-S.P.); yhm1998@163.com (H.-M.Y.)
* Correspondence: yanbinhit@hotmail.com

Received: 30 October 2019; Accepted: 30 November 2019; Published: 5 December 2019

Abstract: In recent years, reversible data hiding (RDH) has become a research hotspot in the field of multimedia security that has aroused more and more researchers' attention. Most of the existing RDH algorithms are aiming at continuous-tone images. For RDH in encrypted halftone images (RDH-EH), the original cover image cannot be recovered losslessly after the watermark is extracted. For some application scenarios such as medical or military images sharing, reversibility is critical. In this paper, a reversible data hiding scheme in encrypted color halftone images (RDH-ECH) is proposed. In the watermark embedding procedure, the cover image is copied into two identical images to increase redundancy. We use wet paper code to embed the watermark into the image blocks. Thus, the receiver only needs to process the image blocks by the check matrices in order to extract the watermarks. To increase embedding capacity, we embed three layers in the embedding procedure and combine the resulting images into one image for convenience of transmission. From the experimental results, it can be concluded that the original image can be restored entirely after the watermarks are extracted. Besides, for marked color halftone images, our algorithm can implement high embedding capacity and moderate visual quality.

Keywords: reversible data hiding; image encryption; color halftone image; wet paper code

1. Introduction

Data hiding refers to making the cover image imperceptibly and keeping the existence of the hidden data confidential. We call the hidden data a *watermark*. Reversible data hiding (RDH) is a technique that the cover image can be restored losslessly after the watermark is extracted [1]. This characteristic makes RDH suitable for high authentication scenarios, such as medical image processing or military communication. It is widely used in applications such as law forensics [2–4], military imagery, medical imagery [5], cloud storage [6,7], copyright protection [8–11], etc. The traditional RDH schemes can be classified into several categories such as lossless compression [12], difference expansion (DE) [13,14], histogram shifting (HS) [15,16], and pixel prediction [17]. Combined with image encryption schemes, many algorithms of RDH in encrypted domain (RDH-ED) have been proposed. According to whether the image decryption and data extraction procedures are separated, the RDH-ED algorithms can be categorized into two types: (1) inseparable algorithms; and (2) separable algorithms.

For inseparable algorithms, Puech et al. [18] applied RDH algorithms to encrypted images by embedding additional bits into pixels blocks encrypted by AES (Advanced Encryption Standard). In 2011, Zhang proposed an algorithm that flips the three least significant bits (LSBs) of half of the pixels in each block [19]. The embedded additional data can be extracted after the encrypted image

with additional data being decrypted. Hong et al. [20] improved Zhang's algorithm [19] by adopting a better measurement method for the smoothness of blocks and using the side-match scheme to decrease the error rate of extraction.

For separable algorithms, Zhang proposed a separable RDH-ED, in which image decryption and data extraction procedures are independent operations [21]. Qian et al. introduced an algorithm that solved a problem of Zhang's scheme [22], which cannot recover the original image completely. In [23], Ma et al. proposed an improved embedding capacity algorithm that needed to reserve room before encryption. Besides, the embedding capacity of the encrypted image is improved. Fu et al. [24] proposed an effective algorithm with adaptive encoding strategy. MSB (most significant bit) layers of embeddable blocks are adaptively compressed according to occurrence frequency of MSB in order to vacate room for data accommodation.

As for halftone images, there are few RDH algorithms available in the literature. Since the redundancy left in halftone image is low, most RDH algorithms for continuous-tone images cannot be directly used in halftone images. Depending on whether the continuous-tone image is available to the embedding process, the RDH algorithms for halftone images can be classified into two categories:

(1) Only the halftone image joins the embedding process. Fu and Oscar proposed RDH-EH algorithm by forced complementary toggling at pseudo-random locations if only the halftone image is available [25]. Lien et al. introduced a high-capacity RDH method for ordered dithered halftone images, which applied dither matrix to pixel pairs so that abundant data can be hidden into these pixel pairs [26]. Kim et al. proposed a separable RDH algorithm, which used Hamming codes to embed watermark [27].
(2) The continuous-tone image joins the embedding process. For the situation in which the original continuous-tone image is available, Fu and Oscar proposed an algorithm that integrates the data hiding operation into the error diffusion process [25]. Lo et al.'s algorithm embedded the binary data into the halftone images with reference to the original continuous-tone image by evaluating the absolute difference between the neighboring gray-level pixels [28]. This method is extended from that of Fu and Au [25].

For the problem of RDH in encrypted halftone image, the remaining redundancy is very difficult to utilize. Designing RDH algorithm for encrypted halftone image is a challenging problem. In this paper, we focus on the topic of RDH in encrypted color halftone images (RDH-ECH), where the original cover image can be restored from the stego image after extracting the watermark. To overcome the problem that in the encrypted halftone images little redundancy exists, we adopt dual cover images in the embedding process. The role of dual images are different from the two shares in visual cryptography [29,30]. Dual images are used to create redundancy for data hiding. We also combine the resulting stego images for facilitating transmission. Because of the reversibility, the proposed method can be applied to data hiding applications such as the healthcare industry and online distribution systems. Unless owning the secret keys, attackers cannot restore the original halftone image and watermark losslessly.

The remaining part of this paper is organized as follows. Section 2 introduces the necessary background of the proposed algorithm which is wet paper code scheme. The proposed scheme is elaborated in Section 3. The experimental results are presented in Section 4, which includes test of lossless recovery, security, visual quality, embedding capacity, and computational complexity. Finally, Section 5 concludes this paper.

Unless otherwise stated, all vectors are column vectors by default.

2. Related Works

In this section, we briefly describe wet paper code (WPC) for RDH. WPC is similar to matrix embedding as both are based on syndrome codes. Let us assume that the sender has a cover binary vector $\boldsymbol{x} = \{x_i\}_{i=1}^{n}$ and a set of indices $Q \subset \{1, \cdots, n\}$ ($|Q| = k$). The k bits are called dry elements

that can be modified to embed watermark, and the remaining $(n-k)$ bits are called wet elements, which remain the same during embedding procedure.

To embed m bits $\boldsymbol{m} \in \{0,1\}^m$, the sender modifies the changeable bits so that the stego binary vector $\boldsymbol{y}$ satisfies

$$\boldsymbol{D} \times \boldsymbol{y} = \boldsymbol{m} \tag{1}$$

where $\boldsymbol{D}$ is an $m \times n$ parity check matrix produced by the pseudo-random number generator. The receiver can extracts the watermark from $\boldsymbol{y}$ by calculating the syndrome $\boldsymbol{Dy}$.

To solve the system of linear equations of Equation (1), we rewrite it as

$$\boldsymbol{D} \times \boldsymbol{v} = \boldsymbol{m} - \boldsymbol{D} \times \boldsymbol{x} \tag{2}$$

where $\boldsymbol{v} = \boldsymbol{y} - \boldsymbol{x}$. There are k unknowns v_j which are nonzeros, whereas the remaining $(n-k)$ values v_i are zeros. Thus, on the left-hand side, we remove from $\boldsymbol{D}$ all $(n-k)$ columns i and also remove from $\boldsymbol{v}$ all $(n-k)$ elements v_i. Keeping the same symbol for $\boldsymbol{v}$, Equation (2) now becomes

$$\boldsymbol{H} \times \boldsymbol{v} = \boldsymbol{m} - \boldsymbol{D} \times \boldsymbol{x} \tag{3}$$

where $\boldsymbol{H}$ is a binary $m \times k$ matrix consisting of those columns of $\boldsymbol{D}$, and $\boldsymbol{v}$ is an unknown binary $k \times 1$ vector holding the embedding changes. According to the solvability of linear equations, the value of $\boldsymbol{v}$ is solved. Thus, we get the modified binary column vector $\boldsymbol{y}$ based on $\boldsymbol{y} = \boldsymbol{v} + \boldsymbol{x}$.

3. Proposed Scheme

In this section, we elaborate on our RDH scheme for encrypted color halftone image. The block diagram of our method is illustrated in Figure 1.

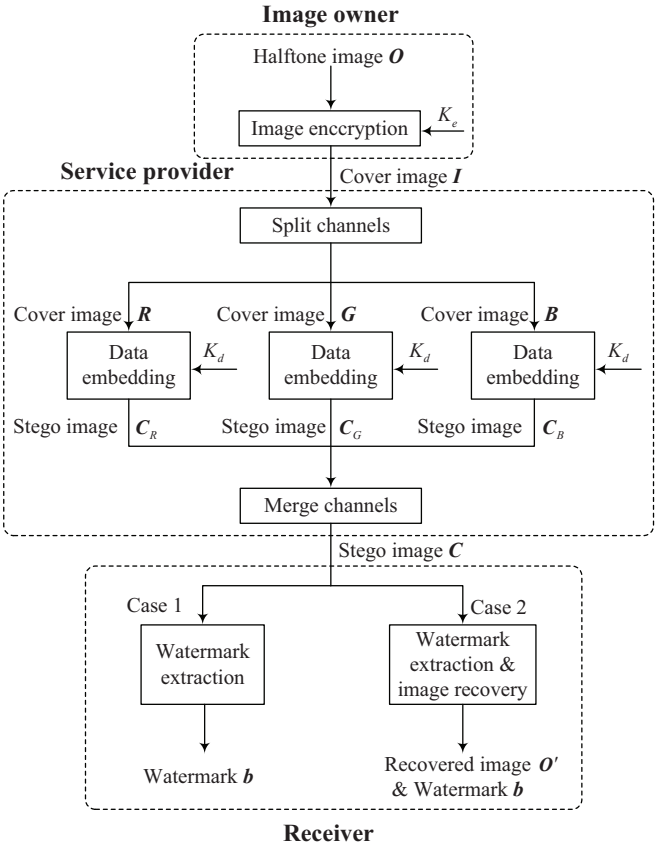


Figure 1. Block diagram of the proposed scheme.

The overall procedure of proposed RDH-EH algorithm can be briefly described as follows. First, the image owner encrypts the original halftone image $\boldsymbol{O}$ using encryption key K_e and then sends the produced cover image $\boldsymbol{I}$ to the service provider. The provider first converts $\boldsymbol{I}$ into channel images $\boldsymbol{R}$, $\boldsymbol{G}$, and $\boldsymbol{B}$. For cover image $\boldsymbol{R}$, he copies it into two identical cover image $\boldsymbol{R}'$ and $\boldsymbol{R}''$. After the first WPC embedding procedure is complete, two stego image $\boldsymbol{F}_{R'}$ and $\boldsymbol{F}_{R''}$ are generated. The stego image $\boldsymbol{F}_{R'}$ is the cover image of the second WPC embedding process and is copied into two identical images $\boldsymbol{F}'_{R'}$ and $\boldsymbol{F}''_{R'}$. There will be stego images $\boldsymbol{S}_{R'}$ and $\boldsymbol{S}_{R''}$ after the second WPC. Next, the service provider uses $\boldsymbol{S}_{R'}$ in the third embedding procedure and also copies it into two images $\boldsymbol{S}'_{R'}$ and $\boldsymbol{S}''_{R'}$. After that, two stego images $\boldsymbol{T}_{R'}$ and $\boldsymbol{T}_{R''}$ are produced. Then, the provider combines $\boldsymbol{T}_{R'}$, $\boldsymbol{F}_{R''}$, $\boldsymbol{S}_{R''}$, and $\boldsymbol{T}_{R''}$ into one image $\boldsymbol{C}_R$; the specific way of combining them is elaborated in Section 3.2. For cover images $\boldsymbol{G}$ and $\boldsymbol{B}$, be executing the same embedding process as described above, the stego images $\boldsymbol{C}_G$ and $\boldsymbol{C}_B$ are generated. Finally, the service provider merges stego images $\boldsymbol{C}_R$, $\boldsymbol{C}_G$, and $\boldsymbol{C}_B$ into one stego image $\boldsymbol{C}$.

On the receiver's side, depending on the limits of authority, he can do different operations to the image. If the receiver has only the data-hiding key K_d, he can extract the watermark $\boldsymbol{b}$. If he has both K_e and K_d, the receiver can extract the watermark $\boldsymbol{b}$ and recover the original image $\boldsymbol{O}$. Because we combine four images into one stego image during the embedding procedure, if the receiver has only the K_e, he does not have access to the content of the original image after decryption.

3.1. Image Encryption

Suppose that the size of the original color halftone image $\boldsymbol{O}$ is $N_1 \times N_2 \times 3$. As each pixel in a halftone image is represented by one bit, the total number of bits N is $N_1 \times N_2 \times 3$. Let the pixel value at position (u, v, t) be $a_{u,v,t}$, where $\{1 \leq u \leq N_1\}$, $\{1 \leq v \leq N_2\}$, and $\{1 \leq t \leq 3\}$. To encrypt $\boldsymbol{O}$, we calculate exclusive-or operations between $a_{u,v,t}$ and the pseudo-random bits $k_{u,v,t}$ as

$$e_{u,v,t} = a_{u,v,t} \oplus k_{u,v,t} \tag{4}$$

where $e_{u,v,t}$ is the pixel value of cover image $\boldsymbol{I}$, $k_{u,v,t}$ is generated by a stream cipher scheme with encryption key K_e. For example, one may use any secure stream ciphers such as RC4 [31].

3.2. Data Embedding

In this subsection, we describe the watermark embedding procedure which adopts dual images. The diagram of data embedding procedure is illustrated in Figure 2. Given the cover image $\boldsymbol{I}$, the service provider first splits the cover image into $\boldsymbol{R}$, $\boldsymbol{G}$, and $\boldsymbol{B}$ channels and does the same operation to each channel as follows.

(1) Taking cover image $\boldsymbol{R}$ as an example, for the first WPC embedding, the service provider first copies it into two identical cover images $\boldsymbol{R}'$ and $\boldsymbol{R}''$. We partition each of them into non-overlapping blocks of size $\alpha \times \beta$; each block is denoted as $\boldsymbol{L}_{\ell j}$, $j = 1, \cdots, w$ and $\ell = 1, 2$, where w is the total number of blocks for each channel image:

$$w = \left\lfloor \frac{N_1}{\alpha} \right\rfloor \times \left\lfloor \frac{N_2}{\beta} \right\rfloor \tag{5}$$

where $\lfloor x \rfloor$ rounds x to the nearest integer towards $-\infty$.

The service provider first encrypts the watermark $b_{\gamma i}$ by

$$\theta_{\gamma i} = b_{\gamma i} \oplus p_{\gamma i} \tag{6}$$

where $\gamma = 1, \cdots, 9$, because we perform the embedding process three times on one channel image and there are three channels, $\theta_{\gamma i}$ is the encrypted watermark, and $p_{\ell i}$ is determined via standard

stream cipher using data-hiding key K_d. Then, for every four bits of each $\theta_{\gamma i}$, we group them to a block. Thus, the jth watermark block $\boldsymbol{\theta}_{\gamma i}$ is obtained as

$$\boldsymbol{\theta}_{\gamma j} = [\theta_{\gamma(4j-3)}\ \theta_{\gamma(4j-2)}\ \theta_{\gamma(4j-1)}\ \theta_{\gamma(4j)}]^T,\ 1 \le j \le w \tag{7}$$

For cover image $\boldsymbol{R}'$, we read one block $\boldsymbol{L}_{1j}$, and convert it into a column vector $\boldsymbol{x}_{1j}$. Let the first four elements of $\boldsymbol{x}_{1j}$ be the dry elements of WPC. Generate the pseudo-random binary matrix $\boldsymbol{D}_{1j}$ of dimensions $q \times n$ by using K_d. Then, to generate the column vector $\boldsymbol{y}_{1j}$, which satisfies

$$\boldsymbol{D}_{1j} \times \boldsymbol{y}_{1j} = \boldsymbol{\theta}_{1j} \tag{8}$$

we need first calculate $\boldsymbol{s}_{1j} = \boldsymbol{D}_{1j} \times \boldsymbol{x}_{1j}$ according to Equation (3). Next, the exclusive-or operations between $\boldsymbol{s}_{1j}$ and $\boldsymbol{\theta}_{1j}$ are implemented. After that, the last four columns of each $\boldsymbol{D}_{1j}$ are removed to form $\boldsymbol{H}_{1j}$. Finally, $\boldsymbol{v}_{1j}$ is solved based on the solvability of the linear equations and then

$$\boldsymbol{y}_{1j} = \boldsymbol{v}_{1j} + \boldsymbol{x}_{1j} \tag{9}$$

Having $\boldsymbol{y}_{1j}$, new blocks $\boldsymbol{L}'_{1j}$ are formed and, then, these blocks are combined to produce stego image $\boldsymbol{F}_{R'}$ by following raster scanning order.

As for cover image $\boldsymbol{R}''$, we embed each watermark vector $\boldsymbol{\theta}_{1j}$ into $\boldsymbol{L}_{2j}$. First, the first four bits of each block $\boldsymbol{L}_{2j}$ are converted to a vector $\boldsymbol{z}_{1j}$. Then, embed watermark vector into $\boldsymbol{z}_{1j}$ by

$$\bar{\boldsymbol{z}}_{1j} = \boldsymbol{z}_{1j} \oplus \boldsymbol{\theta}_{1j} \tag{10}$$

After all blocks of $\boldsymbol{R}''$ are modified, we obtain new blocks $\boldsymbol{L}'_{2j}$ and also stego image $\boldsymbol{F}_{R''}$.

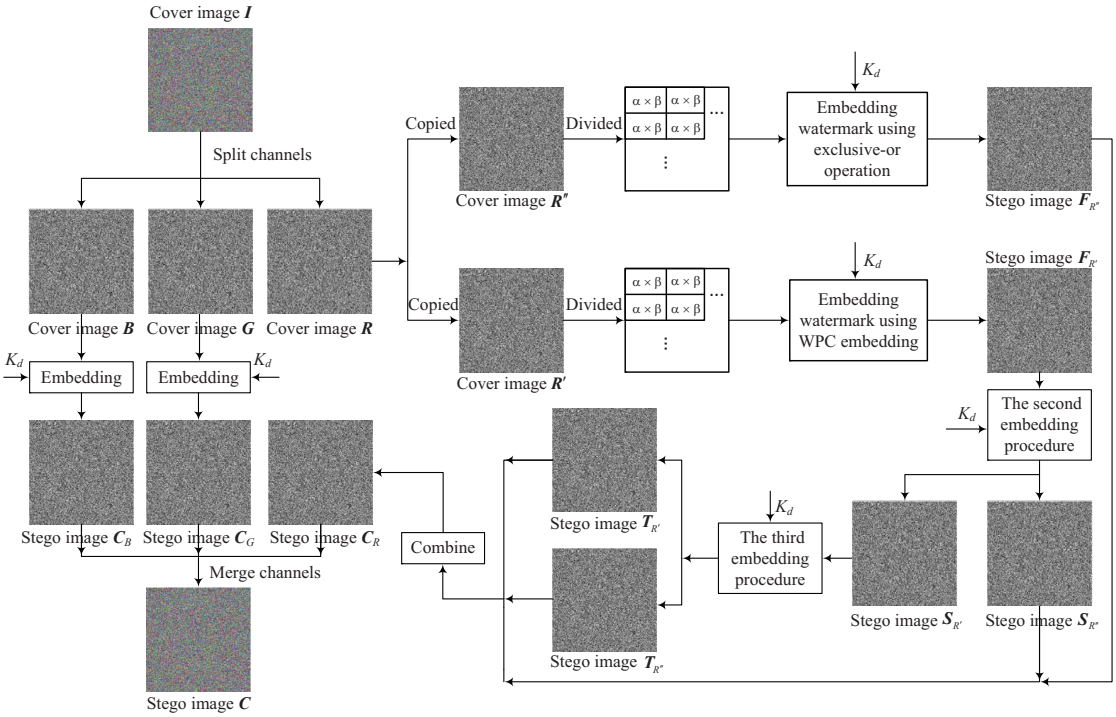


Figure 2. Block diagram of data embedding procedure.

(2) For the second WPC embedding, we take image $\boldsymbol{F}_{R'}$ as the cover image. The same as the above procedure, the service provider first copies $\boldsymbol{F}_{R'}$ into two identical images $\boldsymbol{F}'_{R'}$ and $\boldsymbol{F}''_{R'}$. For cover image $\boldsymbol{F}'_{R'}$, we take the same WPC embedding operation as in Step (1) and then generate stego image $\boldsymbol{S}_{R'}$. For cover image $\boldsymbol{F}''_{R'}$, we embed watermark vector in it and then we get stego image $\boldsymbol{S}_{R''}$.

(3) As for the third WPC embedding, we take image $\boldsymbol{S}_{R'}$ as the cover image and copy it into two same images $\boldsymbol{S}'_{R'}$ and $\boldsymbol{S}''_{R'}$. After the third embedding procedure, stego images $\boldsymbol{T}_{R'}$ and $\boldsymbol{T}_{R''}$ are produced.

(4) Finally, we combine four images $\boldsymbol{T}_{R'}$, $\boldsymbol{F}_{R''}$, $\boldsymbol{S}_{R''}$, and $\boldsymbol{T}_{R''}$ into one image $\boldsymbol{C}_R$ using K_d. As demonstrated in Figure 3, supposing that each image is 2×2 sized and the pixels are represented by different color, we stagger-stitch these pixels up–down and left–right.

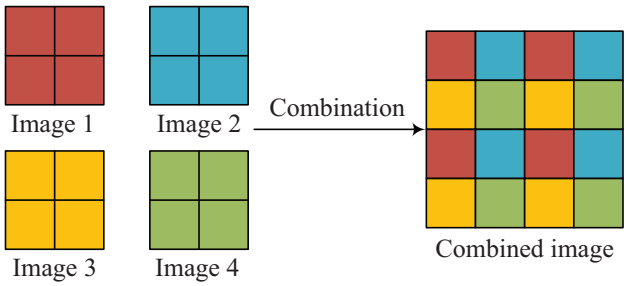


Figure 3. The demonstration of image combination.

The reason why we adopt triple embedding is that it produces four stego images after embedding. Thus, the aspect ratio of the combined image is the same as the original image $\boldsymbol{O}$. The different aspect ratio of the combined image for different levels of embedding is listed in Table 1. The comparison result shows that triple embedding is suitable for image combination as it keeps the aspect ratio of the original image.

Table 1. Aspect ratios resulting from different embedding levels.

Embedding Levels	Number of Result Images	Aspect Ratio
Zero	1	N_1:N_2
Once	2	$2N_1$:N_2
Twice	3	$3N_1$:N_2
Triple	4	N_1:N_2

For cover image $\boldsymbol{G}$ and $\boldsymbol{B}$, we perform the same embedding procedure to them, and then we have combined stego images $\boldsymbol{C}_G$ and $\boldsymbol{C}_B$. Finally, we merge the three channel images $\boldsymbol{C}_R$, $\boldsymbol{C}_G$, and $\boldsymbol{C}_B$ and then the stego image $\boldsymbol{C}$ is generated. The proposed data embedding procedure is summarized in Algorithm 1.

Algorithm 1 Data embedding.

Input: Cover image $\boldsymbol{I}$, watermark $\boldsymbol{b}$, and data-hiding key K_d
Output: Stego image $\boldsymbol{C}$

1: Split cover image $\boldsymbol{I}$ into channel images $\boldsymbol{R}$, $\boldsymbol{G}$, and $\boldsymbol{B}$
2: **function** TRIPLEEMBEDDING($\boldsymbol{I}, \boldsymbol{b}, K_d$)
3: Encrypt watermark $\boldsymbol{b}_i$ to θ_i using Equation (6)
4: Group θ_i to block $\boldsymbol{\theta}_j$
5: Copy $\boldsymbol{I}$ into $\boldsymbol{I}'$ and $\boldsymbol{I}''$
6: For $\boldsymbol{I}'$ do:
7: Divide $\boldsymbol{I}'$ into blocks $\boldsymbol{L}_{1j}$
8:
9: **for** $j = 1$ to w **do**
10: $\boldsymbol{x}_j \leftarrow \boldsymbol{L}_{1j}$
11: Generate $\boldsymbol{D}_j$ using K_d
12: $\boldsymbol{s}_j = \boldsymbol{D}_j \times \boldsymbol{x}_j$
13: $\bar{\boldsymbol{s}}_j = \boldsymbol{s}_j \oplus \boldsymbol{\theta}_j$
14: $\boldsymbol{H}_j \leftarrow \boldsymbol{D}_j$
15: $\boldsymbol{v}_j \leftarrow (\boldsymbol{H}_j \boldsymbol{v}_j = \bar{\boldsymbol{s}}_j)$
16: $\boldsymbol{y}_j = \boldsymbol{v}_j + \boldsymbol{x}_j$
17: Blocks $\boldsymbol{L}'_{1j} \leftarrow \boldsymbol{y}_j$
18: **end for**
19: Stego image $\boldsymbol{F}_{I'} \leftarrow \boldsymbol{L}'_{1j}$
20: For $\boldsymbol{I}''$ do:
21: Divided $\boldsymbol{I}''$ into blocks $\boldsymbol{L}_{2j}$
22:
23: **for** $j = 1$ to w **do**
24: Assign $\boldsymbol{z}_j$ with the first four bits of $\boldsymbol{L}_{2j}$
25: $\bar{\boldsymbol{z}}_j = \boldsymbol{z}_j \oplus \boldsymbol{\theta}_j$
26: Assign $\bar{\boldsymbol{z}}_j$ to the first four bits of $\boldsymbol{L}'_{2j}$
27: **end for**
28: Stego image $\boldsymbol{F}_{I''} \leftarrow \boldsymbol{L}'_{2j}$ //The first embedding process ends
29: For $\boldsymbol{F}_{I'}$ do:
30: Repeat Steps 5–19, generate stego image $\boldsymbol{S}_{I'}$
31: Repeat Steps 20–28, generate stego image $\boldsymbol{S}_{I''}$
32: //The second embedding process ends
33: For $\boldsymbol{S}_{I'}$ do:
34: Repeat Steps 5–19, generate stego image $\boldsymbol{T}_{I'}$
35: Repeat Steps 20–28, generate stego image $\boldsymbol{T}_{I''}$
36: //The third embedding process ends
37: $\boldsymbol{C}_I$ combined from $\boldsymbol{T}_{I'}$, $\boldsymbol{F}_{I''}$, $\boldsymbol{S}_{I''}$, and $\boldsymbol{T}_{I''}$
38: **end function**
39: $\boldsymbol{C}_R$ = TRIPLEEMBEDDING($\boldsymbol{R}, \boldsymbol{b}, K_d$)
40: $\boldsymbol{C}_G$ = TRIPLEEMBEDDING($\boldsymbol{G}, \boldsymbol{b}, K_d$)
41: $\boldsymbol{C}_B$ = TRIPLEEMBEDDING($\boldsymbol{B}, \boldsymbol{b}, K_d$)
42: Merge channel images $\boldsymbol{C}_R$, $\boldsymbol{C}_G$ and $\boldsymbol{C}_B$ to color image $\boldsymbol{C}$
43: **return** Stego image $\boldsymbol{C}$

3.3. Extraction and Recovery

In this section, we discuss the two situations on the receiver's side. Depending on the different types of key, the receiver can extract the watermark (having K_d) or recover the image after extraction (having both K_e and K_d). Because we combine the images into one stego image, if we only decrypt stego image $\boldsymbol{C}$, the content of the original image cannot be directly presented. The proposed procedure of extraction and recovery is summarized in Algorithm 2.

Algorithm 2 Extraction and recovery.

Input: Stego image $\boldsymbol{C}$, secret key K_e and K_d
Output: Original halftone image $\boldsymbol{O}$ and watermark $b_{\ell i}$

Split stego image $\boldsymbol{O}$ into channel images $\boldsymbol{C}_R$, $\boldsymbol{C}_G$, and $\boldsymbol{C}_B$
function DATAEXTRACTION($\boldsymbol{C}, K_d$)
    Separate $\boldsymbol{C}$ to four images $\boldsymbol{T}_{C'}$, $\boldsymbol{F}_{C''}$, $\boldsymbol{S}_{C''}$, and $\boldsymbol{T}_{C''}$
    Divide $\boldsymbol{T}_{C'}$ into blocks $\boldsymbol{W}_{1j}$
    Divide $\boldsymbol{T}_{C''}$ into blocks $\boldsymbol{W}_{2j}$

    for $j = 1$ to w **do**
        $\boldsymbol{y}'_{3j} \leftarrow \boldsymbol{W}_{1j}$
        Generated $\boldsymbol{D}_{3j}$ using K_d
        $\boldsymbol{\theta}_{3j} = \boldsymbol{D}_{3j} \times \boldsymbol{y}'_{3j}$
        Assign $\bar{z}_{3j}$ to the first four bits of $\boldsymbol{W}_{2j}$
        $\boldsymbol{z}'_{3j} = \bar{z}_{3j} \oplus \boldsymbol{\theta}_{3j}$
    end for
    Stego image $\boldsymbol{S}_{C'} \leftarrow \boldsymbol{z}'_{3j}$
    Divide $\boldsymbol{S}_{C'}$ into blocks $\boldsymbol{E}_{1j}$
    Divide $\boldsymbol{S}_{C''}$ into blocks $\boldsymbol{E}_{2j}$

    for $j = 1$ to w **do**
        $\boldsymbol{y}'_{2j} \leftarrow \boldsymbol{E}_{1j}$
        Generated $\boldsymbol{D}_{2j}$ using K_d
        $\boldsymbol{\theta}_{2j} = \boldsymbol{D}_{2j} \times \boldsymbol{y}'_{2j}$
        Assign $\bar{z}_{2j}$ to the first four bits of $\boldsymbol{E}_{2j}$
        $\boldsymbol{z}'_{2j} = \bar{z}_{2j} \oplus \boldsymbol{\theta}_{2j}$
    end for
    Stego image $\boldsymbol{F}_{C'} \leftarrow \boldsymbol{z}'_{2j}$
    Divide $\boldsymbol{F}_{C'}$ into blocks $\boldsymbol{P}_{1j}$
    Divide $\boldsymbol{F}_{C''}$ into blocks $\boldsymbol{P}_{2j}$

    for $j = 1$ to w **do**
        $\boldsymbol{y}'_{1j} \leftarrow \boldsymbol{P}_{1j}$
        Generated $\boldsymbol{D}_{1j}$ using K_d
        $\boldsymbol{\theta}_{1j} = \boldsymbol{D}_{1j} \times \boldsymbol{y}'_{1j}$
        Assign $\bar{z}_{1j}$ to the first four bits of $\boldsymbol{P}_{2j}$
        $\boldsymbol{z}'_{1j} = \bar{z}_{1j} \oplus \boldsymbol{\theta}_{1j}$
    end for
    Cover image $\leftarrow \boldsymbol{z}'_{1j}$
end function
$(\boldsymbol{R}, \boldsymbol{\theta}_{1j}, \boldsymbol{\theta}_{2j}\boldsymbol{\theta}_{3j}) =$ DATAEXTRACTION$(\boldsymbol{C}_R, K_d)$
$(\boldsymbol{G}, \boldsymbol{\theta}_{4j}, \boldsymbol{\theta}_{5j}, \boldsymbol{\theta}_{6j}) =$ DATAEXTRACTION$(\boldsymbol{C}_G, K_d)$
$(\boldsymbol{B}, \boldsymbol{\theta}_{7j}, \boldsymbol{\theta}_{8j}, \boldsymbol{\theta}_{9j}) =$ DATAEXTRACTION$(\boldsymbol{C}_B, K_d)$
$b_{\ell i} = \theta_{\ell i} \oplus p'_{\ell i}$
Merge images $\boldsymbol{R}$, $\boldsymbol{G}$ and $\boldsymbol{B}$ to cover image $\boldsymbol{I}$
Decrypt $\boldsymbol{I}$ using Equation(16)
return Watermark $b_{\ell i}$ and original image $\boldsymbol{O}$

3.3.1. Watermark Extraction

When the receiver has only the data-hiding key K_d, he can extract the watermark from stego image $\boldsymbol{C}$. The receiver first divides the image into channel images $\boldsymbol{C}_R$, $\boldsymbol{C}_G$, and $\boldsymbol{C}_B$. For stego image $\boldsymbol{C}_R$, he separates it into four images $\boldsymbol{T}_{R'}$, $\boldsymbol{F}_{R''}$, $\boldsymbol{S}_{R''}$, and $\boldsymbol{T}_{R''}$. Stego images $\boldsymbol{T}_{R'}$ and $\boldsymbol{T}_{R''}$ are divided into $\alpha \times \beta$ non-overlapping blocks, each of which is denoted as $\boldsymbol{W}_{\ell j}$, $j = 1, \cdots, w$ and $\ell = 1, 2$. Convert

each of $\boldsymbol{W}_{1j}$ to a column vector $\boldsymbol{y}'_{3j}$, and, the following equation is calculated to obtain the encrypted watermark blocks $\boldsymbol{\theta}_{3j}$ according to Equation (8)

$$\boldsymbol{\theta}_{3j} = \boldsymbol{D}_{3j} \times \boldsymbol{y}'_{3j} \tag{11}$$

where $\boldsymbol{D}_{3j}$ is pseudo-random binary matrix generated by K_d. Then, read the first four bits of $\boldsymbol{W}_{2j}$ and convert them to a column vector $\bar{\boldsymbol{z}}_{3j}$. Next, replace the first four bits of every block by

$$\begin{aligned} \boldsymbol{z}'_{3j} &= \bar{\boldsymbol{z}}_{3j} \oplus \boldsymbol{\theta}_{3j} \\ &= \boldsymbol{z}_{3j} \oplus \boldsymbol{\theta}_{3j} \oplus \boldsymbol{\theta}_{3j} \\ &= \boldsymbol{z}_{3j} \end{aligned} \tag{12}$$

After that, stego image $\boldsymbol{S}_{R'}$ is acquired. We divide $\boldsymbol{S}_{R'}$ and $\boldsymbol{S}_{R''}$ into 1×8 non-overlapping blocks, each of which is denoted as $\boldsymbol{E}_{\ell j}$, $j = 1, \cdots, w$ and $\ell = 1, 2$. Convert each of $\boldsymbol{E}_{1j}$ into vectors $\boldsymbol{y}'_{2j}$, and then into the encrypted watermark blocks $\boldsymbol{\theta}_{2j}$ by calculating

$$\boldsymbol{\theta}_{2j} = \boldsymbol{D}_{2j} \times \boldsymbol{y}'_{2j} \tag{13}$$

Next, the first four bits of each block of $\boldsymbol{E}_{1j}$ are converted to column vectors $\bar{\boldsymbol{z}}_{2j}$. Then, replace them by

$$\boldsymbol{z}'_{2j} = \bar{\boldsymbol{z}}_{2j} \oplus \boldsymbol{\theta}_{2j} \tag{14}$$

Thus, stego image $\boldsymbol{F}_{R'}$ is achieved, and we also get cover image $\boldsymbol{R}$ and encrypted watermark blocks $\boldsymbol{\theta}_{1j}$ by performing the same process using $\boldsymbol{F}_{R'}$ and $\boldsymbol{F}_{R''}$. In addition, cover images $\boldsymbol{G}$ and $\boldsymbol{B}$ as well as watermark blocks $\boldsymbol{\theta}_{\gamma' j}$ ($\gamma' = 4, \cdots, 9$) are also generated from stego images $\boldsymbol{C}_G$ and $\boldsymbol{C}_B$. We concatenate $\boldsymbol{\theta}_{\gamma j}$ to column vectors $\boldsymbol{\theta}_{\gamma i}$ and get watermark $b_{\gamma i}$ by calculating

$$b_{\gamma i} = \theta_{\gamma i} \oplus p'_{\gamma i} \tag{15}$$

where $\gamma = 1, \cdots, 9$, $p'_{\gamma i}$ is determined via standard stream cipher using data-hiding key K_d. Finally, images $\boldsymbol{R}$, $\boldsymbol{G}$ and $\boldsymbol{B}$ are merged to compose cover image $\boldsymbol{I}$.

3.3.2. Image Recovery

If the receiver has both K_e and K_d, he can recover the original halftone image after watermark extraction. By decrypting the cover image $\boldsymbol{I}$ we get in Section 3.3.1, the original image $\boldsymbol{O}$ is generated. Let the pixel value at position (u, v, t) be $a'_{u,v,t}$, where $\{1 \le u \le N_1\}$, $\{1 \le v \le N_2\}$, and $\{1 \le t \le 3\}$. To decrypt $\boldsymbol{I}$, we calculate exclusive-or operations between $a'_{u,v,t}$ and the pseudo-random bits $k'_{u,v,t}$ as

$$e'_{u,v,t} = a'_{u,v,t} \oplus k'_{u,v,t} \tag{16}$$

where $e'_{u,v,t}$ is the pixel value of original image $\boldsymbol{O}$, $k'_{u,v,t}$ is generated by the same stream cipher scheme as in the image encryption phase using K_e. The diagram of watermark extraction and image recovery is demonstrated in Figure 4.

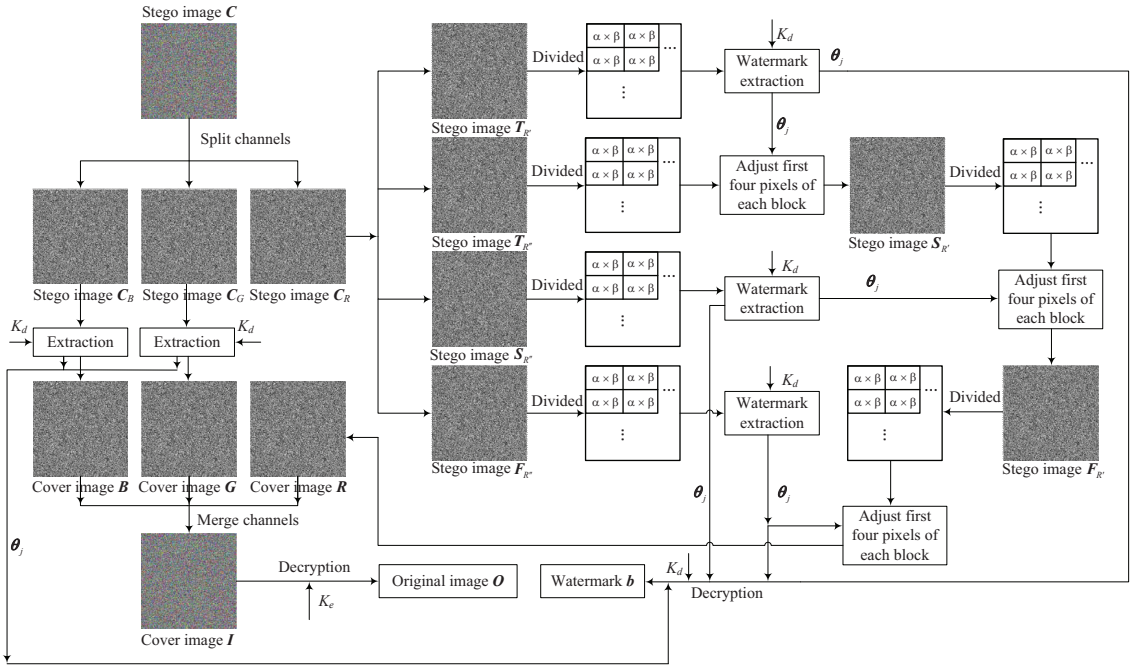


Figure 4. The diagram of watermark extraction and image recovery.

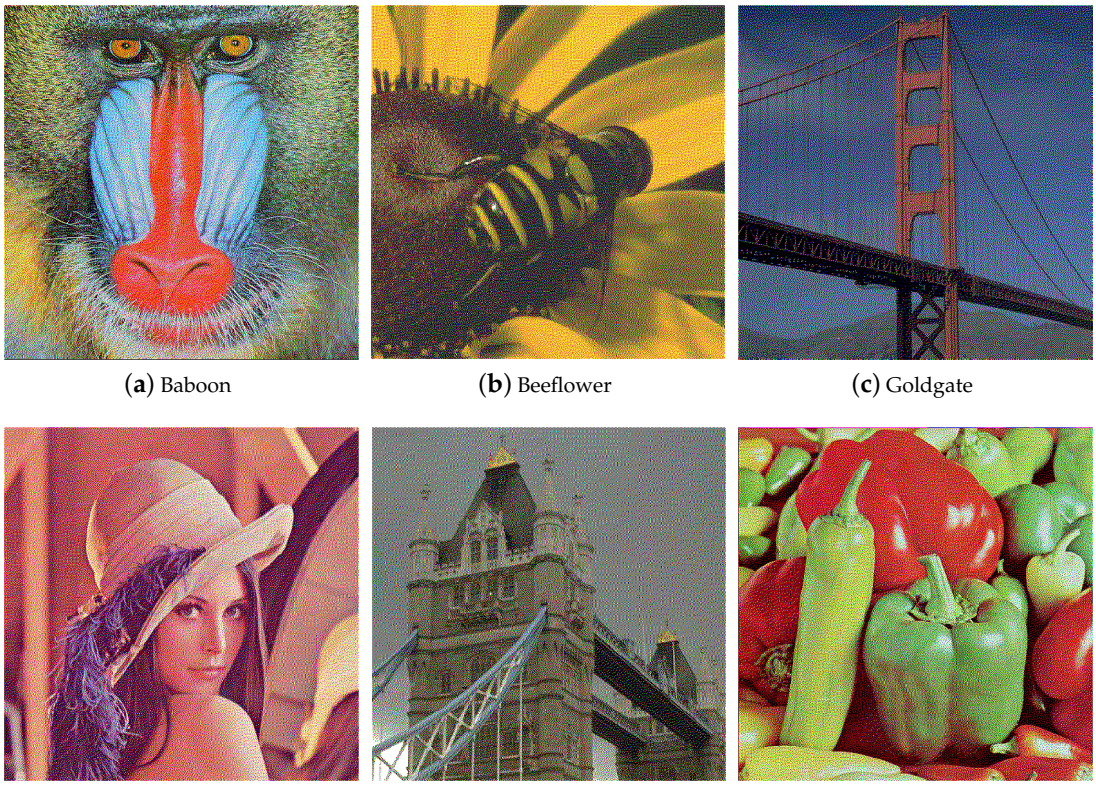
4. Experimental Results and Discussions

To demonstrate the effectiveness and superiority of our scheme, experiments were conducted on six test images sized 512 × 512: Baboon, Beeflower, Goldgate, Lena, London, and Peppers (see Figure 5). The choice of free parameters is tabulated in Table 2. For simplicity of calculation, α was set to 1 and β was set to 8. Thus, $n = \alpha \times \beta = 8$. q was 4 because we embedded four bits to each block. The environment of our experiments was based on a personal computer with a 3.2 GHz Intel i5 processor, 4 GB memory, Windows 10 operating system, and Matlab R2016a. In the following subsections, five aspects of experimental results are presented: (1) reversibility; (2) security; (3) embedding capacity; (4) visual quality; and (5) computational complexity. Finally, we give the feature comparisons with related schemes.

To view the result image of each procedure intuitively, we take Figure 5b as an example. Figure 6a shows the original halftone image, and Figure 6b is the corresponding encrypted result $\boldsymbol{I}$. We can observe that principal contents of original image $\boldsymbol{O}$ were effectively masked after encryption. Figure 6c shows the stego image $\boldsymbol{C}$, which is embedded with watermark and displays at 50%. Figure 6d is the recovered image $\boldsymbol{O}'$, which is exactly the same as the original image $\boldsymbol{O}$.

Table 2. Selection of free parameters.

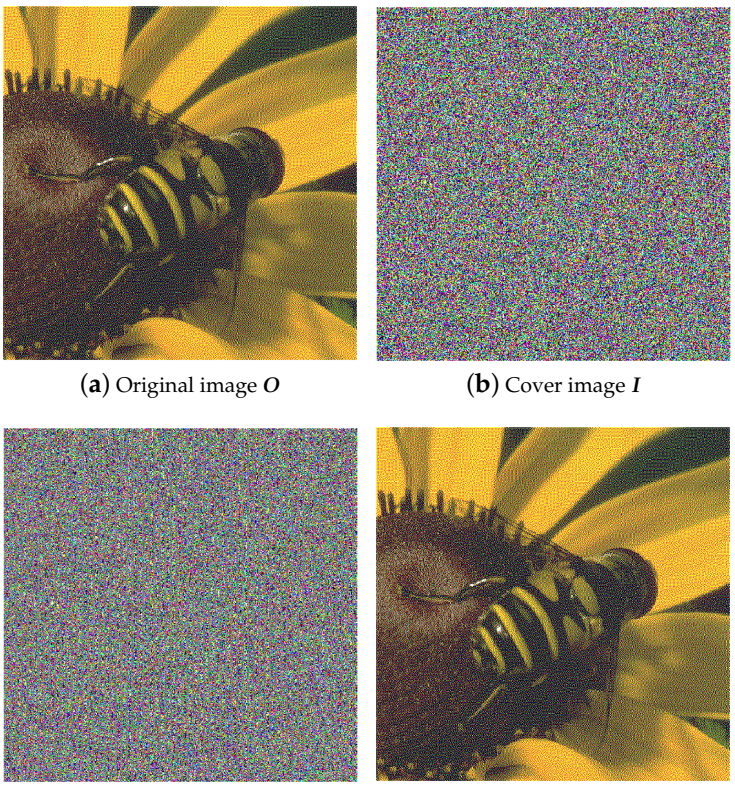
Parameter	α	β	q	n
Value	1	8	4	8



(**a**) Baboon (**b**) Beeflower (**c**) Goldgate

(**d**) Lena (**e**) London (**f**) Peppers

Figure 5. Original image O: (**a**) Baboon; (**b**) Beeflower; (**c**) Goldgate; (**d**) Lena; (**e**) London; and (**f**) Peppers.



(**a**) Original image O (**b**) Cover image I

(**c**) Stego image C(Displays at 50%) (**d**) Recovered image O'

Figure 6. The whole images of every procedure with Beeflower of blocks size 1×8.

4.1. Reversibility

The reversibility of an RDH scheme is that the original image can be recovered losslessly from the stego image. In such case, the scheme is reversible. As shown in Section 3.3, our method is a reversible one. To verify whether the recovered image $\boldsymbol{O}'$ is identical to the original image $\boldsymbol{O}$, we adopted the correlation coefficient r and MSE (mean squared error). The correlation coefficient is widely used in statistical analysis of the correlation between two variables:

$$r(\boldsymbol{O}', \boldsymbol{O}) = \frac{\sum_m \sum_n (O'_{m,n} - \bar{O}')(O_{m,n} - \bar{O})}{\sqrt{(\sum_m \sum_n (O'_{m,n} - \bar{O}')^2)(\sum_m \sum_n (O_{m,n} - \bar{O})^2)}} \tag{17}$$

where $O'_{m,n}$ and $O_{m,n}$ are pixel values at the (m, n) position of recovered image $\boldsymbol{O}'$ and original image $\boldsymbol{O}$, respectively. $\bar{O}'$ and $\bar{O}$ are the mean values of $\boldsymbol{O}'$ and $\boldsymbol{O}$, respectively. The value of r is limited in the range $[-1, 1]$. The closer the value is to 1, the more similar the two images are to each other. Therefore, the value of r is supposed to be 1 if the two images are identical. The MSE between two images $\boldsymbol{O}'$ and $\boldsymbol{O}$ is defined as

$$MSE(\boldsymbol{O}', \boldsymbol{O}) = \frac{1}{N_1 N_2} \sum_{m=1}^{N_1} \sum_{n=1}^{N_2} (O'_{m,n} - O_{m,n})^2 \tag{18}$$

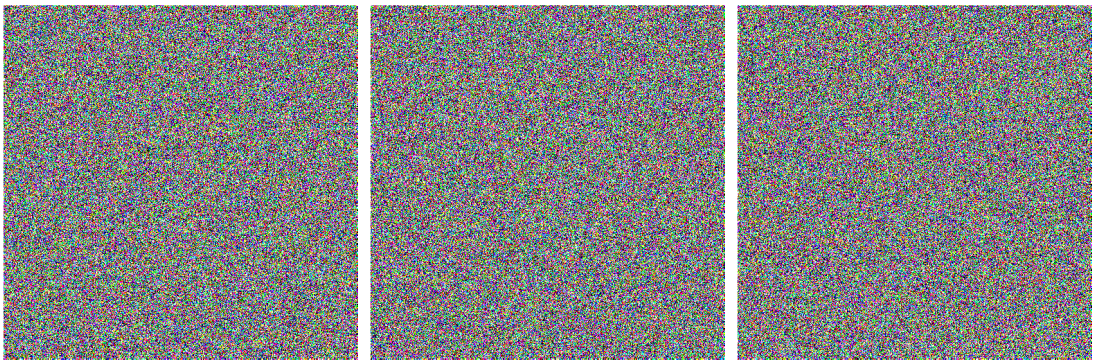
which is zero if $\boldsymbol{O}'$ and $\boldsymbol{O}$ are identical. The correlation coefficients of original images and recovered images are all 1 s and all the MSE values are 0 s. That is to say, our scheme is reversible.

4.2. Security

The original images and the watermark are encrypted by the stream cipher scheme. As shown in Figure 7, the encrypted images are completely meaningless and it is difficult to infer their original images from them. To assess the security quantitatively, we used correlation coefficient r (Equation (17)) and peak signal-to-noise ratio (PSNR), which is widely used in evaluating the quality of digital images. The PSNR between cover image $\boldsymbol{I}$ and original halftone image $\boldsymbol{O}$ is defined via the mean square error (MSE) by

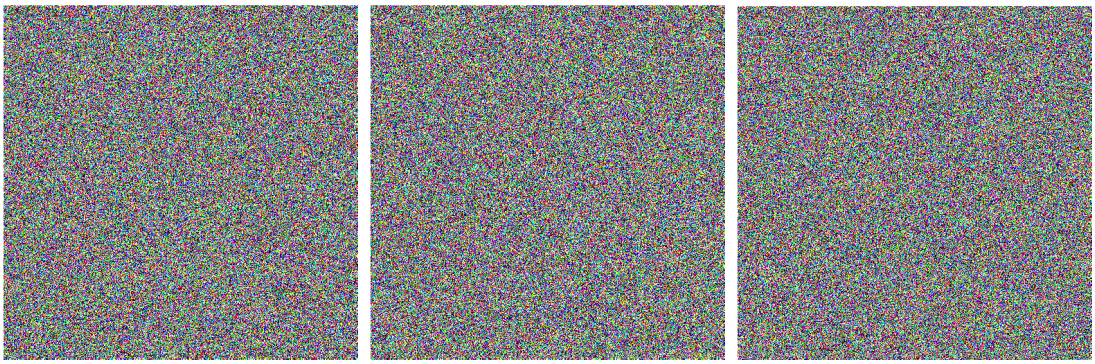
$$PSNR(\boldsymbol{O}, \boldsymbol{I}) = 10 \cdot \log_{10} \frac{M_O^2}{MSE(\boldsymbol{O}, \boldsymbol{I})} \text{ (dB)} \tag{19}$$

where M_O is the maximum possible pixel value of $\boldsymbol{O}$. It is 1 since the pixels of halftone images are represented using 1 bit per sample. The calculation results are consistent with the presentation of Figure 7.



(**a**) Baboon (**b**) Beeflower (**c**) Goldgate

Figure 7. *Cont.*



(**d**) Lena (**e**) London (**f**) Peppers

Figure 7. Cover image $\boldsymbol{I}$ of: (**a**) Baboon; (**b**) Beeflower; (**c**) Goldgate; (**d**) Lena; (**e**) London; and (**f**) Peppers.

4.3. Embedding Capacity

To evaluate the embedding capacity, the embedding rate E_r was adopted:

$$E_r = \frac{N_e}{N_t} \tag{20}$$

where N_e and N_t are the number of embedded bits and the total bits of the original halftone image, respectively. We divided the six test images into blocks of the same size of 1×8, and we embedded 4 bits into each block. The embedding capacity E_c of these images was the same:

$$E_c = w \times 4 \times 3 \times 3 = \lfloor \frac{512}{1} \rfloor \times \lfloor \frac{512}{8} \rfloor \times 4 \times 3 \times 3 = 1,179,648 \text{ (bits)} \tag{21}$$

where w is the total amount of blocks for each channel image, we embedded three times on each channel image and there were three channel images. Table 3 shows various embedding capacity when several different sizes of blocks were employed. Lower embedding capacity may lead to a higher visual quality of the marked image. If we want to know the content of the image and we directly decrypt the image, the higher is the image quality, the clearer is the image content.

Table 3. Embedding capacity of different sizes of blocks.

Block Size	Block Amount	Size of m	E_c	E_r
1×8	98,304	4	1,179,648	1.5
1×8	98,304	2	589,824	0.75
4×4	49,152	3	442,368	0.5625
8×8	12,288	20	737,280	0.9375
16×16	3072	32	294,912	0.375
16×16	3072	16	147,456	0.1875
16×16	3072	8	73,728	0.0938

4.4. Quality of the Marked Image

We decrypted the stego image directly and then obtained marked image $\boldsymbol{M}$ (see examples in Figure 8). The different PSNR values of marked images are shown in Figure 9a for the six test images under different embedding capacity. Besides, the structural similarity index (SSIM) was also utilized to evaluate the quality in Figure 9b

$$SSIM(\boldsymbol{M}, \boldsymbol{O}) = \frac{(2\mu_M\mu_O + C_1)(2\sigma_{MO} + C_2)}{(\mu_M^2 + \mu_O^2 + C_1)(\sigma_M^2 + \sigma_O^2 + C2)} \tag{22}$$

where μ_M, μ_O, σ_M, σ_O, and σ_{MO} are the local means, standard deviations, and cross-covariance for images $\boldsymbol{M}$ and $\boldsymbol{O}$, respectively.

It can be found in the comparison results that, when we adopt a lower capacity such as 73728 bits, the quality of the marked image $\boldsymbol{M}$ will be better.

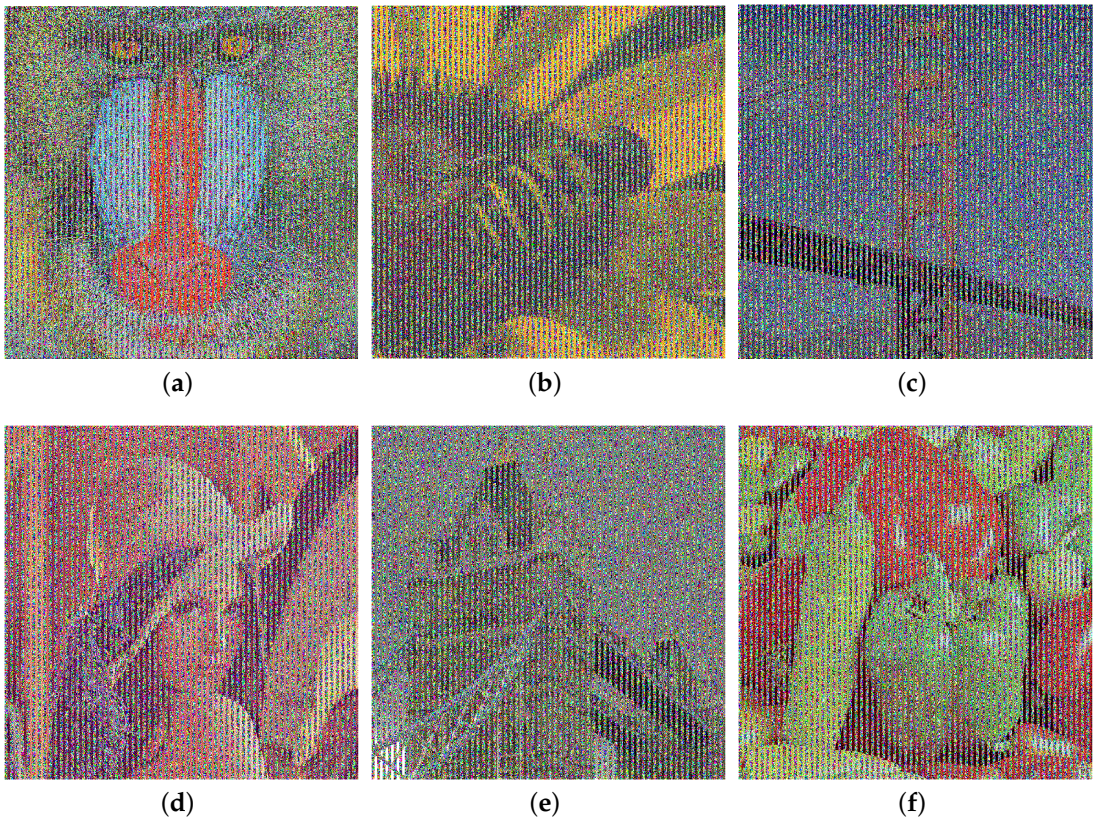
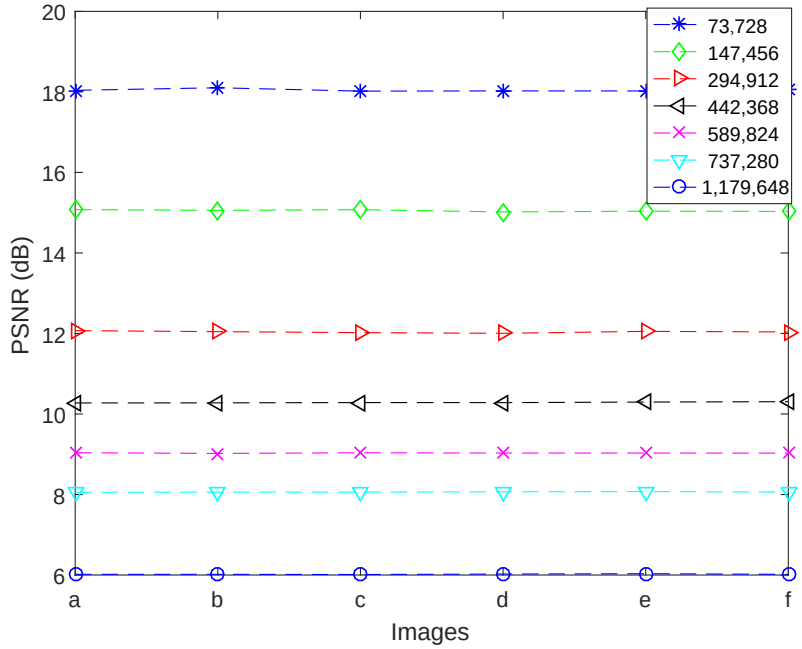
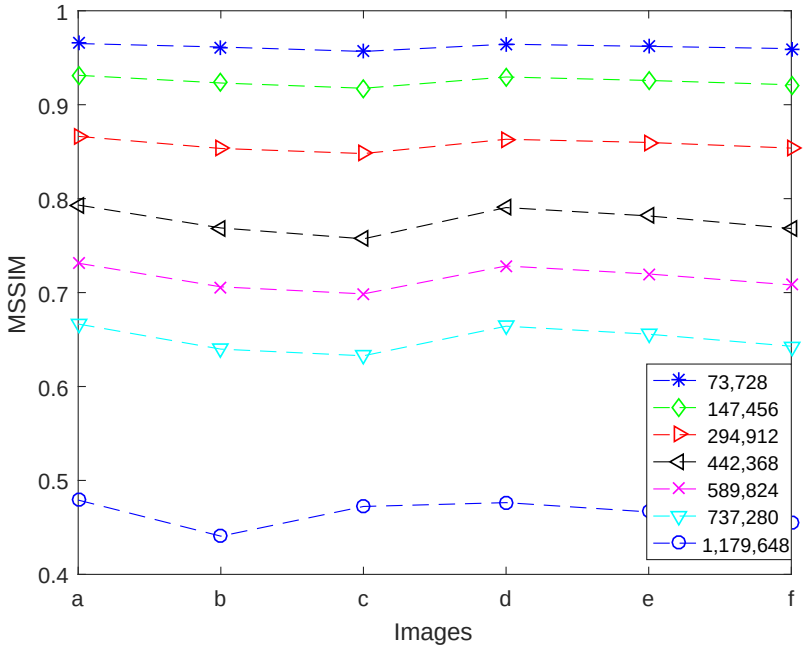


Figure 8. Examples of marked images $\boldsymbol{M}$ of: (**a**) Baboon; (**b**) Beeflower; (**c**) Goldgate; (**d**) Lena; (**e**) London; and (**f**) Peppers.



(**a**) PSNR of test images with capacity: 73,728, 147,456, 294,912, 442,368, 589,824, 737,280, and 1,179,648 bits.



(**b**) SSIM of test images with capacity: 73,728, 147,456, 294,912, 442,368, 589,824, 737,280, and 1,179,648 bits.

Figure 9. Comparison of image quality with different embedding capacity: (**a**) image quality in terms of PSNR; and (**b**) image quality in terms of SSIM.

4.5. Computational Complexity

For the proposed scheme, the computational complexity of encrypting and decrypting pixels is $\mathcal{O}(N)$ ($\mathcal{O}$ notation is used to classify algorithms according to how their running time or space requirements grow as the input size grows [32].) because they are linear-time algorithms and the

running time increases at most linearly with the size N of the image. The computational complexity of embedding and extraction procedure is $\mathcal{O}(mn^2)$ according to Equation (1), where m and n are the length of the embedded bits of each block and the length of each block, respectively.

4.6. Feature Comparisons

In Table 4, feature comparisons among the proposed method and other related methods are given. Different from other methods, the carrier images of our method are color halftone images. In addition, our method has real reversibility and provides a high embedding capacity for halftone images, although less redundancy exists in encrypted color halftone images.

Table 4. Feature comparisons among the proposed method and related methods

Method	Real Reversible	Embedding Capacity (bits)	Image Type	Application Domain
Zhang [19]	No	Below 4400 [23]	Grayscale continuous-tone	Encrypt
Zhang [21]	Yes	4400	Grayscale continuous-tone	Encrypt
Ma et al. [23]	Yes	131,072	Grayscale continuous-tone	Encrypt
Fu et al. [24]	Yes	416,809	Grayscale continuous-tone	Encrypt
Lien et al. [33]	Yes	79,438	Grayscale halftone	Plaintext
Chen et al. [34]	No	23,814	Grayscale halftone	Plaintext
Jia et al. [35]	Yes	56,617	Grayscale continuous-tone	Plaintext
Kim et al. [27]	No	4096	Grayscale halftone	Encrypt
Li et al. [36]	Yes	725,000	Color continuous-tone	Plaintext
Ours	Yes	1,179,648	Color halftone	Encrypt

5. Conclusions

In this paper, a reversible data hiding scheme for encrypted color halftone images is proposed. As less information redundancy is available for halftone images, we adopted two stego images to add redundancy artificially. This can increase the embedding capacity of the scheme, and, more importantly, the original image can be restored. Moreover, we combine the four stego images generated in embedding procedure into one image for easy transmission. By using two secret keys, the cover image can be recovered losslessly after the watermarks are extracted. Experimental results demonstrate the advantages of our methods. The proposed scheme can realize real reversibility with high embedding capacity for encrypted color halftone images.

Author Contributions: Conceptualization, Y.-X.S. and B.Y.; software, Y.-X.S. and B.Y.; Formal analysis, Y.-X.S., B.Y., and J.-S.P.; Methodology, Y.-X.S., B.Y., and H.-M.Y.; Writing—original draft, Y.-X.S., B.Y., J.-S.P., and N.C.; and Writing—review and editing, B.Y., J.-S.P., H.-M.Y., and N.C.

Funding: This work was funded by the National Natural Science Foundation of China (NSFC) (No. 61272432), Shandong Provincial Natural Science Foundation (No. ZR2014JL044), and MOE (Ministry of Education in China) Project of Humanities and Social Sciences (Project No. 18YJAZH110).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Cox, I.; Miller, M.; Bloom, J.; Fridrich, J.; Kalker, T. *Digital Watermarking and Steganography*; Morgan Kaufmann: San Francisco, CA, USA, 2007.
2. Chen, C.; Xiang, B.; Liu, Y.; Wang, K. A Secure Authentication Protocol for Internet of Vehicles. *IEEE Access* **2019**, *7*, 12047–12057. [CrossRef]
3. Wu, T.Y.; Chen, C.M.; Wang, F.; Meng, C.; Wang, E.K. A provably secure certificateless public key encryption with keyword search. *J. Chin. Inst. Eng.* **2019**, *42*, 1–9. doi:10.1080/02533839.2018.1537807. [CrossRef]
4. Pan, J.S.; Lee, C.Y.; Sghaier, A.; Zeghid, M.; Xie, J. Novel Systolization of Subquadratic Space Complexity Multipliers Based on Toeplitz Matrix-Vector Product Approach. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2019**, *27*, 1614–1622. [CrossRef]

5. Zhang, W.; Ma, K.; Yu, N. Reversibility improved data hiding in encrypted images. *Signal Process.* **2014**, *94*, 118–127. [CrossRef]
6. Xia, Z.; Wang, X.; Zhang, L.; Qin, Z.; Sun, X.; Ren, K. A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing. *IEEE Trans. Inf. Forensics Secur.* **2016**, *11*, 2594–2608. [CrossRef]
7. Qian, Z.; Xu, H.; Luo, X.; Zhang, X. New framework of reversible data hiding in encrypted JPEG bitstreams. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *29*, 351–362. [CrossRef]
8. Shi, H.; Liu, D.; Lu, H.; Zhou, C. A homomorphic encrypted reversible information hiding scheme for integrity authentication and piracy tracing. *Multimed. Tools Appl.* **2018**, *77*, 20535–20567. [CrossRef]
9. Yang, C.H.; Tsai, M.H. Improving histogram-based reversible data hiding by interleaving predictions. *IET Image Process.* **2010**, *4*, 223–234. [CrossRef]
10. Luo, H.; Pan, T.; Pan, J.; Chu, S.; Yang, B. Development of a Three-Dimensional Multimode Visual Immersive System With Applications in Telepresence. *IEEE Syst. J.* **2017**, *11*, 2818–2828. [CrossRef]
11. Weng, S.; Chen, Y.; Ou, B.; Chang, C.; Zhang, C. Improved K-Pass Pixel Value Ordering Based Data Hiding. *IEEE Access* **2019**, *7*, 34570–34582. doi:10.1109/ACCESS.2019.2904174. [CrossRef]
12. Fridrich, J.; Goljan, M.; Du, R. Lossless data embedding for all image formats. In *Security and Watermarking of Multimedia Contents IV*; International Society for Optics and Photonics: Orlando, FL, USA, 2002; Volume 4675, pp. 572–583.
13. Weng, S.; Pan, J.s.; Li, L. Reversible data hiding based on an adaptive pixel-embedding strategy and two-layer embedding. *Inf. Sci.* **2016**, *369*, 144–159. [CrossRef]
14. Tian, J. Reversible data embedding using a difference expansion. *IEEE Trans. Circuits Syst. Video Technol.* **2003**, *13*, 890–896. [CrossRef]
15. Du, Y.; Yin, Z.; Zhang, X. Improved lossless data hiding for jpeg images based on histogram modification. *Comput. Mater. Contin.* **2018**, *55*, 495–507.
16. Tai, W.L.; Yeh, C.M.; Chang, C.C. Reversible data hiding based on histogram modification of pixel differences. *IEEE Trans. Circuits Syst. Video Technol.* **2009**, *19*, 906–910.
17. Luo, L.; Chen, Z.; Chen, M.; Zeng, X.; Xiong, Z. Reversible image watermarking using interpolation technique. *IEEE Trans. Inf. Forensics Secur.* **2009**, *5*, 187–193.
18. Puech, W.; Chaumont, M.; Strauss, O. A reversible data hiding method for encrypted images. In *Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*; International Society for Optics and Photonics: Orlando, FL, USA, 2008; Volume 6819, p. 68191E.
19. Zhang, X. Reversible data hiding in encrypted image. *IEEE Signal Process. Lett.* **2011**, *18*, 255–258. [CrossRef]
20. Hong, W.; Chen, T.S.; Wu, H.Y. An improved reversible data hiding in encrypted images using side match. *IEEE Signal Process. Lett.* **2012**, *19*, 199–202. [CrossRef]
21. Zhang, X. Separable reversible data hiding in encrypted image. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 826–832. [CrossRef]
22. Qian, Z.; Zhang, X.; Wang, S. Reversible data hiding in encrypted JPEG bitstream. *IEEE Trans. Multimed.* **2014**, *16*, 1486–1491. [CrossRef]
23. Ma, K.; Zhang, W.; Zhao, X.; Yu, N.; Li, F. Reversible data hiding in encrypted images by reserving room before encryption. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 553–562. [CrossRef]
24. Fu, Y.; Kong, P.; Yao, H.; Tang, Z.; Qin, C. Effective reversible data hiding in encrypted image with adaptive encoding strategy. *Inf. Sci.* **2019**, *494*, 21–36. [CrossRef]
25. Fu, M.S.; Au, O.C. Data hiding watermarking for halftone images. *IEEE Trans. Image Process.* **2002**, *11*, 477–484.
26. Lien, B.K.; Lin, Y.M.; Lee, K.Y. High-capacity reversible data hiding by Maximum-span pixel Pairing on ordered dithered halftone images. In Proceedings of the 2012 IEEE 19th International Conference on Systems, Signals and Image Processing (IWSSIP), Vienna, Austria, 11–13 April 2012; pp. 76–79.
27. Kim, C.; Shin, D.; Leng, L.; Yang, C.N. Separable reversible data hiding in encrypted halftone image. *Displays* **2018**, *55*, 71–79. [CrossRef]
28. Lo, C.C.; Lee, C.M.; Liao, B.Y.; Pan, J.S. Halftone Image Data Hiding with Reference to Original Multitone Image. In Proceedings of the 2008 IEEE International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Harbin, China, 15–17 August 2008; pp. 265–268.
29. Naor, M.; Shamir, A. Visual cryptography. In *Workshop on the Theory and Application of of Cryptographic Techniques*; Springer: Berlin/Heidelberg, Germany, 1994; pp. 1–12.

30. Yan, B.; Xiang, Y.; Hua, G. Improving the Visual Quality of Size-Invariant Visual Cryptography for Grayscale Images: An Analysis-by-Synthesis (AbS) Approach. *IEEE Trans. Image Process.* **2019**, *28*, 896–911. [CrossRef]
31. Stallings, W. The RC4 stream encryption algorithm. In *Cryptography and Network Security*; Prentice Hall: Upper Saddle River, NJ, USA, 2005.
32. Mohr, A. *Quantum Computing in Complexity Theory and Theory of Computation*; Business in Calgary: Carbondale, IL, USA, 2014.
33. Lien, B.K.; Lin, Y.m. High-capacity reversible data hiding by maximum-span pairing. *Multimed. Tools Appl.* **2011**, *52*, 499–511. [CrossRef]
34. Chen, Y.Y.; Chen, W.S. High-quality blind watermarking in halftones using random toggle approach. *Multimed. Tools Appl.* **2018**, *77*, 8019–8041. [CrossRef]
35. Jia, Y.; Yin, Z.; Zhang, X.; Luo, Y. Reversible data hiding based on reducing invalid shifting of pixels in histogram shifting. *Signal Process.* **2019**, *163*, 238–246. [CrossRef]
36. Li, J.; Li, X.; Yang, B. Reversible data hiding scheme for color image based on prediction-error expansion and cross-channel correlation. *Signal Process.* **2013**, *93*, 2748–2758. [CrossRef]